UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/688,640	10/17/2003	Antti Kokkinen	14322US02	1958

23446 7590 07/18/2008
MCANDREWS HELD & MALLOY, LTD
500 WEST MADISON STREET
SUITE 3400
CHICAGO, IL 60661

EXAMINER	
WANG, BEN C	

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
07/18/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/688,640	KOKKINEN, ANTTI	
	Examiner	Art Unit	
	BEN C. WANG	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *18 January 2008*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-24* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-24* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Applicant's amendment dated April 17, 2008, responding to the Office action mailed January 18, 2008 provided in the rejection of claims 1-24, wherein claims 1 and 12 were amended.

Claims 1-24 remain pending in the application and which have been fully considered by the examiner.

Applicant's arguments with respect to claims currently amended have been fully considered but are moot in view of the new grounds of rejection – see *Waldin, JR. et al.* and *Gu et al.* - arts made of record, as applied hereto.

Claim Rejections – 35 USC § 102(e)

The following is quotation of 35 U.S.C. 102(e) which form the basis for all obviousness rejections set forth in this office action:

> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

2. Claims 1-20 are rejected under 35 U.S.C. 102(e) as being anticipated by Waldin, JR. et al. (Pub. No. US 2003/0177485 A1) (hereinafter 'Waldin' – art made of record)

3. **As to claim 1** (Currently Amended), Waldin discloses a method for updating software in an electronic device , the method comprising:

- generating an update package for updating at least one software application being generated based upon difference information between the at least one software application and at least one reference software installed on the electronic device (e.g., Abstract – Lines 5-11- Any version of an application may be upgraded to any other version of the application, through the use of a series of increment update patches (i.e., an update package) ...; Fig. 1; Fig. 6, element 404 - Delta Catalog (i.e., reference software); [0027], Lines 4-19 - ... an incremental update builder, such as binary patch file builder, to produce at least one incremental update, such as binary patch, which can transform a previous version of the software application to the current version ...)
- updating the at least one software application using the update package and the reference software (e.g., [0036], Lines 12-18 – a *DeltaCatalog* corresponds to each increment update. Each *DeltaCatalog* has an associated source state and an associated destination state, and specifies the necessary update information by specifying which *DeltaPackage* (i.e., an update package) should be used by each flavor of the application to update from the source state to the destination state); and
- wherein the updating leaves the at least one reference software unchanged (e.g., [0044], Lines 4-8 – After the requested *DeltaCatalog* is received, the *DeltaCatalog* is processed ... to determine an incremental set of *DeltaPackages* ...; [0041], Lines 15-17 – The *DeltaUpdater* builds a "*DeltaDirectory*" which is a list of available *DeltaCatalogs* ...)

4. **As to claim 2** (Original) (incorporating the rejection in claim 1), Waldin discloses the method wherein generating an update package for updating the at least one software application based upon the at least one reference software installed on the electronic device comprises:

- accessing a copy of the at least one reference software (e.g., [0044], Lines 4-8 – After the requested *DeltaCatalog* is received, the *DeltaCatalog* is processed … to determine an incremental set of *DeltaPackages* …);
- retrieving a copy of the at least one software application (e.g., [0044], Lines 4-8 – After the requested *DeltaCatalog* is received, the *DeltaCatalog* is processed … to determine an incremental set of *DeltaPackages* …); and
- generating an update package (e.g., [0044], Lines 4-8 – After the requested *DeltaCatalog* is received, the *DeltaCatalog* is processed … to determine an incremental set of *DeltaPackages* …)

5. **As to claim 3** (Original) (incorporating the rejection in claim 1), Waldin discloses the method wherein generating an update package for updating at least one software application based upon the at least one reference software installed on the electronic device comprises:

- accessing a copy of the at least one reference software (e.g., [0044], Lines 4-8 – After the requested *DeltaCatalog* is received, the *DeltaCatalog* is processed … to determine an incremental set of *DeltaPackages* …);

- retrieving a copy of each of multiple versions of the at least one software application (e.g., **[0036]**, Lines 12-18 – a *DeltaCatalog* corresponds to each increment update. Each *DeltaCatalog* has an associated source state and an associated destination state, and specifies the necessary update information by specifying which *DeltaPackage* (i.e., an update package) should be used by each flavor of the application to update from the source state to the destination state); and
- generating an update package comprising all transitions between the retrieved versions of the at least one software application (e.g., Abstract – Lines 5-11- Any version of an application may be upgraded to any other version of the application, through the use of a series of increment update patches (i.e., an update package) ...)

6. **As to claim 4** (Original) (incorporating the rejection in claim 1), Waldin discloses the method further comprising updating multiple update versions of the at least one software application installed on the electronic device is performed using a single update package (e.g., Fig. 2 - Δ AJ – update from version A to version J)

7. **As to claim 5** (Original) (incorporating the rejection in claim 1), Waldin discloses the method further comprising installing the at least one software application and the at least one reference software as part of a single installation (e.g., Abstract – Lines 5-11- Any version of an application may be upgraded to any other version of the application,

through the use of a series of increment update patches (i.e., an update package) …; Fig. 1; Fig. 6, elements 404 - Delta Catalog (i.e., reference software); [0027], Lines 4-19 - … an incremental update builder, such as binary patch file builder, to produce at least one incremental update, such as binary patch, which can transform a previous version of the software application to the current version …)

8. **As to claim 6** (Original) (incorporating the rejection in claim 1), Waldin discloses the method further comprising updating the at least one reference software and updating the at least one software application (e.g., Abstract – Lines 5-11- Any version of an application may be upgraded to any other version of the application, through the use of a series of increment update patches (i.e., an update package) …; Fig. 1; Fig. 6, elements 404 - Delta Catalog (i.e., reference software); [0027], Lines 4-19 - … an incremental update builder, such as binary patch file builder, to produce at least one incremental update, such as binary patch, which can transform a previous version of the software application to the current version …) as part of a single update (e.g., Fig. 2 - Δ AJ – update from version A to version J).

9. **As to claim 7** (Original) (incorporating the rejection in claim 1), Waldin discloses the method wherein the at least one software application comprises a plurality of software applications (e.g., Abstract – Lines 5-11- Any version of an application may be upgraded to any other version of the application, through the use of a series of increment update patches (i.e., an update package) …; Fig. 1; Fig. 6, elements 404 -

Delta Catalog (i.e., reference software); [0027], Lines 4-19 - … an incremental update builder, such as binary patch file builder, to produce at least one incremental update, such as binary patch, which can transform a previous version of the software application to the current version …), and the at least one reference software comprises a plurality of reference software (e.g., Fig. 7 – element 408 – Delta Directory; [0041], Lines 15-17 – The DeltaUpdater builds a "*DeltaDirectory*" which is a list of available *DeltaCatalogs* …)

10. **As to claim 8** (Original) (incorporating the rejection in claim 7), Waldin discloses the method further comprising:

- identifying a software application needing updating from the plurality of software applications installed on the electronic device (e.g., Fig. 8b, elements 532 – Is Current State = Desired Ending State? yes, 534 – Request Each Delta Package in the Sequential Set);
- identifying whether a reference software corresponding to the software application needing updating is present on the electronic device, wherein if the reference software is not present, then installing the software application and an associated reference software in a single update on the electronic device (e.g., [0044], Lines 4-8 – After the requested *DeltaCatalog* is received, the *DeltaCatalog* is processed … to determine an incremental set of *DeltaPackages* …)

11. **As to claim 9** (Original) (incorporating the rejection in claim 7), Waldin discloses the method further comprising:

- identifying a software application needing updating from the plurality of software applications installed on the electronic device (e.g., Fig. 8b, elements 532 – Is Current State = Desired Ending State? yes, 534 – Request Each Delta Package in the Sequential Set); and
- identifying whether a reference software corresponding to the software application needing updating is present on the electronic device, wherein if the reference software is present (e.g., Fig. 8b – element 528 – Process Delta Catalog to Identify Delta Packages and Append them to the Sequential Set), then
- retrieving an update package for the software application needing updating (e.g., Fig. 8b – element 536 – Receive Each Delta Package in the Sequential Set);
- verifying the update package (e.g., Fig. 8b – element 536 – Receive Each Delta Package in the Sequential Set); and
- installing the update package on the electronic device (e.g., Fig. 8b, element 538 - Apply Each Delta Package in the Sequential Set)

12. **As to claim 10** (Original) (incorporating the rejection in claim 7), Waldin discloses the method further comprising:

- identifying a software application needing updating from the plurality of software applications installed on the electronic device (e.g., Fig. 8b, elements 532 – Is

Current State = Desired Ending State? yes, 534 – Request Each Delta Package in the Sequential Set);

- determining if the update is needed immediately; and
- storing the update until the update is needed immediately (e.g., [0039] - *DeltaDirectories*)

13. **As to claim 11** (Original) (incorporating the rejection in claim 10), Waldin discloses the method wherein when the update is determined to be needed immediately, then

- invoking an update agent to employ at least the stored update package and reference software (e.g., Abstract – Lines 5-11- Any version of an application may be upgraded to any other version of the application, through the use of a series of increment update patches (i.e., an update package) ...; Fig. 1; Fig. 6, elements 404 - Delta Catalog (i.e., reference software); [0027], Lines 4-19 - ... an incremental update builder, such as binary patch file builder, to produce at least one incremental update, such as binary patch, which can transform a previous version of the software application to the current version ...); and
- updating the software application with the update package (e.g., Fig. 8b, element 538 - Apply Each Delta Package in the Sequential Set)

14. **As to claim 12** (Currently Amended), Waldin discloses a system for updating software, the system comprising:

- an electronic device capable of having software installed thereon (e.g., Abstract, Lines 1-4 – A software application is updated to a newer version by means of incremental update patches …);
- a software delivery device for receiving and installing a reference software to the electronic device if the electronic device does not have the reference software previously installed (e.g., [0044], Lines 4-8 – After the requested *DeltaCatalog* is received, the *DeltaCatalog* is processed … to determine an incremental set of *DeltaPackages* …); and
- the software delivery device receiving and delivering at least one update package (e.g., Fig. 8b, element 536 – Receive each Delta Package in the Sequential Set) to the electronic device, wherein the at least one update package is based on differences between at least one application software and the reference software, and the reference software facilitates (e.g., Abstract – Lines 5-11- Any version of an application may be upgraded to any other version of the application, through the use of a series of increment update patches (i.e., an update package) …; Fig. 1; Fig. 6, elements 404 - Delta Catalog (i.e., reference software); [0027], Lines 4-19 - … an incremental update builder, such as binary patch file builder, to produce at least one incremental update, such as binary patch, which can transform a previous version of the software application to the current version …), using the at least one update package, at least one update to the application software installed on the electronic device (e.g., Fig. 8b – element 538 – Apply each Delta Package in the Sequential Set), and wherein the updating leaves the

reference software unchanged (e.g., [0041], Lines 15-17 – The *DeltaUpdater* builds a "*DeltaDirectory*" which is a list of available DeltaCatalogs …)

15. **As to claim 13** (Original) (incorporating the rejection in claim 12), Waldin discloses the system wherein the electronic device further comprises an update agent (e.g., Fig. 6 – element 126 – Delta Updater; [0027], Lines 19-36 – The binary patch files … which makes the patch files available to an updater program (a "client"). The updater program determines what patch files are necessary, retrieves them and applied them to the application to be updated), the update agent being capable of employing the reference software in conjunction with any retrieved update package to generate updated versions of the application software and also being capable of updating a plurality of application software employing reference software associated with each application software (e.g., Fig. 8b, element 534 – Request each Delta Package in the Sequential Set; 536 – Receive each Delta Package in the Sequential Set; 538 – Apply each Delta Package in the Sequential Set)

16. **As to claim 14** (Original) (incorporating the rejection in claim 12), Waldin discloses the system further comprising an update generating system, the update generating system comprising a loader manager, the loader manager:

- managing loading of application software and application software version updates from the software delivery device (e.g., Fig. 6 – element 126 – Delta Updater; [0027], Lines 19-36 – The binary patch files … which makes the patch

files available to an updater program (a "client"). The updater program determines what patch files are necessary, retrieves them and applied them to the application to be updated);

- employing a loader from a loader module (e.g., Fig. 6 – element 126 – Delta Updater; [0027], Lines 19-36 – … The updater program determines what patch files are necessary, retrieves them and applied them to the application to be updated); and
- employing security services to authenticate software being delivered (e.g., [0045] … uses the digital signature to verify that the DeltaPackages are authentic and have not been altered …)

17. **As to claim 15** (Original) (incorporating the rejection in claim 14), Waldin discloses the system wherein the loader manager further comprises an installation agent for installing application software and downloading files from the software delivery device (e.g., Fig. 6 – element 126 – Delta Updater; [0027], Lines 19-36 – The binary patch files … which makes the patch files available to an updater program (a "client"). The updater program determines what patch files are necessary, retrieves them and applied them to the application to be updated)

18. **As to claim 16** (Original) (incorporating the rejection in claim 16), Waldin discloses the system wherein the loader manager is adapted to:

- identify an application software needing updating (e.g., Fig. 8b, element 532 – Is Current State = Desired Ending State?);

- identify whether reference software associated with the application software needing updating exists (e.g., Abstract – Lines 5-11- Any version of an application may be upgraded to any other version of the application, through the use of a series of increment update patches (i.e., an update package) ...; Fig. 1; Fig. 6, elements 404 - Delta Catalog (i.e., reference software); [0027], Lines 4-19 - ... an incremental update builder, such as binary patch file builder, to produce at least one incremental update, such as binary patch, which can transform a previous version of the software application to the current version ...); and
- coordinating an update of the application software and an associated reference software in a single update (e.g., Fig. 6 – element 126 – Delta Updater; [0027], Lines 19-36 – The binary patch files ... which makes the patch files available to an updater program (a "client"). The updater program determines what patch files are necessary, retrieves them and applied them to the application to be updated; Fig. 2 - Δ AJ – update from version A to version J directly)

19. **As to claim 17** (Original) (incorporating the rejection in claim 14), Waldin discloses the system wherein the loader manager is adapted to:

- retrieve the update package (e.g., Fig. 8b – element 534 – Request Each Delta Package in the Sequential Set);
- access contents of the update package (e.g., Fig. 8b – element 536 – Receive Each Delta Package in the Sequential Set); and
- verify the update package (e.g., [0045] ... uses the digital signature to verify that the DeltaPackages are authentic and have not been altered ...)

20. **As to claim 18** (Original) (incorporating the rejection in claim 14), Waldin discloses the system wherein the loader manager is adapted to determine immediacy of a needed update for a particular application software (e.g., Fig. 6 – element 126 – Delta Updater; [0027], Lines 19-36 – The binary patch files ... which makes the patch files available to an updater program (a "client"). The updater program determines what patch files are necessary, retrieves them and applied them to the application to be updated)

21. **As to claim 19** (Original) (incorporating the rejection in claim 12), Waldin discloses the system wherein the software delivery device is one of a server (e.g., Fig. 6 – element 126 – Delta Updater; [0027], Lines 19-36 – The binary patch files are stored on an update data source (a "server") which makes the patch files available to an updater program (a "client")), a CDROM, and a network (e.g., [0040], Lines 9-11 - ... removable disk media, or a network resource)

22. **As to claim 20** (Original) (incorporating the rejection in claim 12), Waldin discloses the system wherein the electronic device is one of a computer, a digital phone, and a digital camera (e.g., Fig. 1, element 116 – User's Computer)

Claim Rejections – 35 USC § 103(a)

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

23. Claims 21-24 are rejected under 35 U.S.C. 103(a) as being unpatentable over Waldin in view of Gu et al. (Pub. No. US 2005/0234997 A1) (hereinafter 'Gu' – art made of record)

24. **As to claim 21** (Original), Waldin discloses a method for updating software in an electronic device the method comprising:

- generating a first update package for updating at least one software application, the first update package being generated based upon difference information between first and second software versions (e.g., [0046], ... "incremental updates." These updates do not contain entire software applications, but rather only that information necessary to transform a given version of a software application to a newer version ... such incremental software updating is binary patching, performed by programs such as RTPatch™ ... A binary patch replaces only those binary bits of a software application which are different in a newer version ... only a small data file including the differences between the two versions ...);

- generating a second update package for updating the at least one software application, the second update package being generated based upon difference information between first and third software versions (e.g., Fig. 2 – see various delta updates, i.e., Δ AB (update from version A to B), Δ AD –update from version A to D etc.);

Further, Waldin discloses any version of an application may be upgraded to any other version of the application, through the use of a series of incremental update patches (e.g., Abstract), but does not explicitly disclose the following:

- generating a third update package for updating the at least one software application, the third update package being generated based upon difference information between the first and second update packages; and
- updating the at least one software application using the third update package

However, in an analogous art of *Byte-Level File Differencing and Updating Algorithms*, Gu discloses the following:

- generating a third update package for updating the at least one software application, the third update package being generated based upon difference information between the first and second update packages (e.g., Fig. 1 – elements 114 - Byte-Level File Differencing Algorithm, 118 - Byte-Level File Updating Algorithm, 116 - Delta File; 110 - Original File; 112 – New File; [0028] - … the electronic files 110 and 112 include software files … data files … are not so limited. Since any type of file can

be regarded as a byte stream, hereafter a file can be described as a byte stream ...; NOTE: for example, Original File (a first update package), New File (a second update package), and Delta File (a third update package)); and

- updating the at least one software application using the third update package (e.g., [0031] - ... this copy of the new file is then used to update the original file hosted on the client device that is targeted for revision or updating ...)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Gu into the Waldin's system to further provide the following in the Waldin system:

- generating a third update package for updating the at least one software application, the third update package being generated based upon difference information between the first and second update packages; and
- updating the at least one software application using the third update package

The motivation is that it would further enhance the Waldin's system by taking, advancing and/or incorporating the Gu's system which offers significant advantages that the delta file provides the differences between the new and the original files in a format that is up to 99% smaller than the new file; thus, one improvement is a reduction in bandwidth required for transmission of the delta file to the client device, reducing

transmission errors in the received file, and increasing file security as once suggested by Gu (e.g., [0032])

25. **As to claim 22** (Original), Waldin discloses a method for updating software in an electronic device, the method comprising:

- generating a first update package for updating at least one software application, the first update package being generated based upon difference information between a first software version and a reference software corresponding to the at least one software application (e.g., Abstract – Lines 5-11- Any version of an application may be upgraded to any other version of the application, through the use of a series of increment update patches (i.e., an update package) …; Fig. 1; Fig. 6, elements 404 - Delta Catalog (i.e., reference software); [0027], Lines 4-19 - … an incremental update builder, such as binary patch file builder, to produce at least one incremental update, such as binary patch, which can transform a previous version of the software application to the current version …);
- generating a second update package for updating the at least one software application, the second update package being generated based upon difference information a second software version and the reference software corresponding to the at least one software application (e.g., Abstract – Lines 5-11- Any version of an application may be upgraded to any other version of the application, through the use of a series of increment update patches (i.e., an update package) …; Fig. 1; Fig. 6, elements 404 - Delta Catalog (i.e., reference

software); [0027], Lines 4-19 - ... an incremental update builder, such as binary patch file builder, to produce at least one incremental update, such as binary patch, which can transform a previous version of the software application to the current version ...);

Further, Waldin discloses any version of an application may be upgraded to any other version of the application, through the use of a series of incremental update patches (e.g., Abstract), but does not explicitly disclose the following:

- generating a third update package for updating the at least one software application, the third update package being generated based upon difference information between the first and second update packages; and
- updating the at least one software application using the third update package

However, in an analogous art of *Byte-Level File Differencing and Updating Algorithms*, Gu discloses the following:

- generating a third update package for updating the at least one software application, the third update package being generated based upon difference information between the first and second update packages (e.g., Fig. 1 – elements 114 - Byte-Level File Differencing Algorithm, 118 - Byte-Level File Updating Algorithm, 116 - Delta File; 110 - Original File; 112 – New File; [0028] - ... the electronic files 110 and 112 include software files ... data files ... are not so limited. Since any type of file can be regarded as a byte stream, hereafter a file can be described as a byte stream ...; NOTE: for example, Original File (a

first update package), New File (a second update package), and Delta File (a third update package)); and

- updating the at least one software application using the third update package (e.g., [0031] - ... this copy of the new file is then used to update the original file hosted on the client device that is targeted for revision or updating ...)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Gu into the Waldin's system to further provide the following in the Waldin system:

- generating a third update package for updating the at least one software application, the third update package being generated based upon difference information between the first and second update packages; and
- updating the at least one software application using the third update package

The motivation is that it would further enhance the Waldin's system by taking, advancing and/or incorporating the Gu's system which offers significant advantages that the delta file provides the differences between the new and the original files in a format that is up to 99% smaller than the new file; thus, one improvement is a reduction in bandwidth required for transmission of the delta file to the client device, reducing transmission errors in the received file, and increasing file security as once suggested by Gu (e.g., [0032])

26. **As to claim 23** (Original), Waldin discloses a system for updating software, the system comprising:

- an electronic device capable of having software installed thereon (e.g., Abstract – Lines 5-11- Any version of an application may be upgraded to any other version of the application, through the use of a series of increment update patches (i.e., an update package) ...);
- a first update package generator for generating update packages based upon difference information between different versions of software (e.g., Fig. 2 – see various delta updates, i.e., Δ AB (update from version A to B), Δ AD –update from version A to D etc.);

Further, Waldin discloses any version of an application may be upgraded to any other version of the application, through the use of a series of incremental update patches (e.g., Abstract), but does not explicitly disclose the following:

- a second update package generator for generating update packages based upon difference information between different update packages; and
- a software delivery device for delivering at least one update package generated based upon difference information between different update packages to the electronic device

However, in an analogous art of *Byte-Level File Differencing and Updating Algorithms*, Gu discloses the following:

- a second update package generator for generating update packages based upon difference information between different update packages (e.g., Fig. 1 – elements 114 - Byte–Level File Differencing Algorithm, 118 - Byte-Level File Updating Algorithm, 116 - Delta File; 110 - Original File; 112 – New File; [0028] - ... the

electronic files 110 and 112 include software files ... data files ... are not so limited. Since any type of file can be regarded as a byte stream, hereafter a file can be described as a byte stream ...; NOTE: for example, Original File (different update package), New File (different update package), and Delta File (a second update package)); and

- a software delivery device for delivering at least one update package generated based upon difference information between different update packages to the electronic device (e.g., [0031] - ... this copy of the new file is then used to update the original file hosted on the client device that is targeted for revision or updating ...)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Gu into the Waldin's system to further provide the following in the Waldin system:

- a second update package generator for generating update packages based upon difference information between different update packages; and
- a software delivery device for delivering at least one update package generated based upon difference information between different update packages to the electronic device

The motivation is that it would further enhance the Waldin's system by taking, advancing and/or incorporating the Gu's system which offers significant advantages that the delta file provides the differences between the new and the original files in a format that is up to 99% smaller than the new file; thus, one improvement is a reduction in

bandwidth required for transmission of the delta file to the client device, reducing transmission errors in the received file, and increasing file security as once suggested by Gu (e.g., [0032])

27. **As to claim 24** (Original), Waldin discloses a system for updating software, the system comprising:

- an electronic device capable of having software installed thereon (e.g., Abstract – Lines 5-11- Any version of an application may be upgraded to any other version of the application, through the use of a series of increment update patches (i.e., an update package) ...);
- a first update package generator for generating update packages based upon difference information between a version of software and a reference software corresponding to at least one software application (e.g., [0046], ... "incremental updates." These updates do not contain entire software applications, but rather only that information necessary to transform a given version of a software application to a newer version ... such incremental software updating is binary patching, performed by programs such as RTPatch™ ... A binary patch replaces only those binary bits of a software application which are different in a newer version ... only a small data file including the differences between the two versions ...);

Further, Waldin discloses any version of an application may be upgraded to any other version of the application, through the use of a series of incremental update patches (e.g., Abstract), but does not explicitly disclose the following:

- a second update package generator for generating update packages based upon difference information between different update packages; and
- a software delivery device for delivering at least one update package generated based upon difference information between different update packages to the electronic device

However, in an analogous art of *Byte-Level File Differencing and Updating Algorithms*, Gu discloses the following:

- a second update package generator for generating update packages based upon difference information between different update packages (e.g., Fig. 1 – elements 114 - Byte–Level File Differencing Algorithm, 118 - Byte-Level File Updating Algorithm, 116 - Delta File; 110 - Original File; 112 – New File; [0028] - ... the electronic files 110 and 112 include software files ... data files ... are not so limited. Since any type of file can be regarded as a byte stream, hereafter a file can be described as a byte stream ...; NOTE: for example, Original File (different update package), New File (different update package), and Delta File (a second update package)); and
- a software delivery device for delivering at least one update package generated based upon difference information between different update packages to the electronic device (e.g., [0031] - ... this copy of the new file is then used to update

the original file hosted on the client device that is targeted for revision or updating ...)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Gu into the Waldin's system to further provide the following in the Waldin system:

- a second update package generator for generating update packages based upon difference information between different update packages; and
- a software delivery device for delivering at least one update package generated based upon difference information between different update packages to the electronic device

The motivation is that it would further enhance the Waldin's system by taking, advancing and/or incorporating the Gu's system which offers significant advantages that the delta file provides the differences between the new and the original files in a format that is up to 99% smaller than the new file; thus, one improvement is a reduction in bandwidth required for transmission of the delta file to the client device, reducing transmission errors in the received file, and increasing file security as once suggested by Gu (e.g., [0032])

Conclusion

28. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is 571-270-

1240. The examiner can normally be reached on Monday - Friday, 8:00 a.m. - 5:00 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Ben C Wang/
Examiner, Art Unit 2192
July 15, 2008

/Tuan Q. Dam/

Supervisory Patent Examiner, Art Unit 2192